RECONOCIMIENTO DE MATRICULAS

Imagen original: “matricula.jpg”

Estructura general del programa más resultado:

```
load Caracteristicas.mat

% TESTING DEL MODELO

X = caracteristicas;
Y = ['0'; '1'; '2'; '3'; '4'; '5'; '6'; '7'; '8'; ...
    '9'; 'B'; 'C'; 'D'; 'F'; 'G'; 'H'; 'J'; 'K'; 'L'; 'M'; ...
    'N'; 'P'; 'R'; 'S'; 'T'; 'V'; 'W'; 'X'; 'Y';'Z'];

Md1 = fitcecoc(X,Y);
Yp = predict(Md1,X);
confusionmat(Y,Yp);

% AISLAR MATRICULA

I = imread('Matricula.jpg');
I= obtenerMatricula(I);
```

```
imshow(I);
```


```
% CALCULAMOS LOS DESCRIPTORES
X1 = calc_caract(I);

% PREDECIMOS EL VALOR DE LOS CARACTERES
matricula = predict(Md1,X1);
fprintf(matricula);
```

```
6194GJZ
```

Como podemos ver, el programa ha confundido el primer 9 con un 6. El resto de caracteres son correctos.

Función busca la matricula en la imagen y devuelve un recorte que contiene la misma y binarizada:

```
function matricula = obtenerMatricula(I)

    I = rgb2gray(I);

    % BINARIZACION DE LA IMAGEN

    N = 100;
    M = 35;
    matricula = colfilt(I, [N M], 'sliding', @movingAverages);
    SE= strel('disk',5);

    matricula=imopen(matricula,SE);

    % BUSQUEDA DE LA MATRICULA

    CC = bwconncomp(matricula);
    RP = regionprops(CC, 'BoundingBox','Image');
    F = struct2cell(RP);
    A = zeros(1, 5);
    for i= 1:size(F,2)
        aux= cell2mat(F(1,i));
        calc= aux(3)/aux(4);
        if(calc > 4.2 && calc < 4.25)
            A= aux;
            A(5)=i;
        end
    end

    % CROP DE LA IMAGEN

    MAT = RP(A(5)).BoundingBox;
    matricula = imcrop(matricula, MAT);

    % LIMPIEZA DE LA MATRICULA

    SE= strel('disk',4);
    mod = matricula > 0;
    mod = imclose(mod,SE);
    mark = zeros(size(mod));
    mark(1,:) = 1;
    mark(:,1) = 1;
    mark(size(matricula,1),:)=1;
    mark(:,size(matricula,2))=1;
    mark = (mark == 1);
    recMod= imreconstruct(mark,not(mod));
    matricula = not(mod) & not(recMod);
end
```

Función para calcular las características (entregada en la sesión 8):

```
function ret = calc_caract(I)
% Separamos la imagen original entre las diferentes componentes connexas
% (caracteres).
CC = bwconncomp(I);

% Obtenemos imagenes binarias separadas de los caracteres
caracter = regionprops(CC, 'Image');

% Calculo de los descriptores de Fourier para cada caracter (Todas las
% matriculas tienen 7 caracteres

FD_0 = FourierDescriptor(caracter(1).Image);
FD_1 = FourierDescriptor(caracter(2).Image);
FD_2 = FourierDescriptor(caracter(3).Image);
FD_3 = FourierDescriptor(caracter(4).Image);
FD_4 = FourierDescriptor(caracter(5).Image);
FD_5 = FourierDescriptor(caracter(6).Image);
FD_6 = FourierDescriptor(caracter(7).Image);


FD_0 = FD_0';
FD_1 = FD_1';
FD_2 = FD_2';
FD_3 = FD_3';
FD_4 = FD_4';
FD_5 = FD_5';
FD_6 = FD_6';

Fourier = [FD_0;FD_1;FD_2;FD_3;FD_4;FD_5;FD_6];


% Calculo de los descriptores basados en la excentricidad.
exc = regionprops(CC,'Eccentricity');
EXC = cell2mat(struct2cell(exc))';

% Calculo del numero de agujeros de las imagenes
eulerNum= regionprops(CC,'EulerNumber');
holes= struct2cell(eulerNum);
holes= cell2mat(holes);

for it= 1:size(holes,2)
    holes(1,it)= abs(holes(1,it) - 1);
end
HOL = holes';
% Calculo de la longitud del mayor eje respecto al perimetro

majorAxis= regionprops(CC,'MajoraxisLength');
perimeter= regionprops(CC, 'Perimeter');

MA = cell2mat(struct2cell(majorAxis));
PE = cell2mat(struct2cell(perimeter));

DIV = MA./PE;
DIVO = DIV';

ret = [DIVO,HOL,EXC,Fourier];
end
```